Efficient and Secure Chaotic S-Box for Wireless Sensor Network

Ghada Zaïbi, Fabrice Peyrard, Abdennaceur Kachouri, Danièle Fournier-Prunaret, Mounir Samet

Efficient and secure chaotic S-Box for wireless sensor network

Ghada Zaibi[1], Fabrice Peyrard[2]*, Abdennaceur Kachouri[1], Danièle Fournier-Prunaret[3] and Mounir Samet[1]

[1] Sfax Engineering School, University of Sfax, Sfax, BP 3038, Tunisia
[2] IRIT/ENSEEIHT, 2 rue Charles Camichel BP 7122, Toulouse Cedex 7, 31071, France
[3] INSA Toulouse, 135 avenue de Rangueil, Toulouse Cedex 4, 31077, France

ABSTRACT

Information security using chaotic dynamics is a novel topic in the wireless sensor network (WSN) research field. After surveying analog and digital chaotic security systems, we give a state of the art of chaotic S-Box design. The substitution tables are nonlinear maps that strengthen and enhance block crypto-systems. This paper deals with the design of new dynamic chaotic S-Boxes suitable for implementation on wireless sensor nodes. Our proposed schemes are classified into two categories: S-Box based on discrete chaotic map with floating point arithmetic (cascading piecewise linear chaotic map and a three-dimensional map) and S-Box based on discrete chaotic map with fixed-point arithmetic (using discretized Lorenz map and logistic–tent map). The security analysis and implementation process on WSN are discussed. The proposed methods satisfy Good S-Box design criteria and exceed the performance of Advanced Encryption Standard static S-Box in some cases. The energy consumption of different proposals and existing chaotic S-Box designs are investigated via a platform simulator and a real WSN testbed equipped with TI MSP430f1611 micro-controller. The simulations and the experimental results show that our proposed S-Box design with fixed-point arithmetic Lorenz map has the lowest energy-consuming profile compared with the other studied and proposed S-Box design.

KEYWORDS

chaos; energy consumption; S-Box; security; wireless sensor network

***Correspondence**

Fabrice Peyrard, IRIT/ENSEEIHT, 2 rue Charles Camichel BP 7122, Toulouse Cedex 7, 31071, France.
E-mail: Fabrice.Peyrard@irit.fr

1. INTRODUCTION

Chaos theory is an interesting field of research dealing with nonlinear, deterministic, and dynamic systems. It is applied to many different domains such as physics, robotics, biology, finance and encryption. The main important characteristics of chaotic systems are the great dependency on initial conditions (ICs) and parameters variation [1], ergodicity, the random-like behavior, and simplicity. These properties attract the researchers to develop chaotic secure communication and crypto-systems. There are two fundamental methods in modeling chaotic crypto-systems: analog and digital [2]. The first one is based on synchronization between two crypto-chaotic systems in a noisy environment [3]. The second one does not depend on synchronization, which is an important asset. In many digital chaotic crypto-systems, constructing good dynamic substitution boxes (S-Boxes) instead of static ones is the main issue [4–11]. The S-Box is the only nonlinear component in block ciphers and the principal source of confusion property needed to conceive strong crypto-systems. All the proposed approaches are studying the security and robustness of chaotic dynamic S-Boxes. However, to the best of our knowledge, there is no recent study dealing with the pertinence of such an approach for wireless sensor network (WSN) security, and there are only some researches on designing digital chaotic ciphers for WSNs. A WSN is a network composed of active sensor nodes that have small volume, restricted storage space, and restricted communication bandwidth with insufficient battery power. The small nodes are supposed to monitor, manipulate, and transmit gathered insecure environmental information. The security aspect is an emerging research challenge driven by the limited battery resources. Therefore, designing an encryption algorithm for WSN must take into account energy consumption. The contribution of this paper is to propose new dynamic S-Box approaches suitable for the WSN, and to compare the new and existing methods' security aspect and

energy consumption. In this paper, we present a brief overview of the analog and digital chaotic systems, in Section 2, as well as chaotic algorithms conceived for WSN and gives chaotic dynamic S-Box design art state. Our proposed algorithms for S-Box design are presented in Section 3. In Section 4, we provide security analysis and comparison of the different S-Boxes performances.

In Section 5, we adapt and implement the generated S-boxes on a WSN simulator. In Section 6, we evaluate the energy consumption of the different algorithms via simulations and real measurements on an experimental sensor SensLAB testbed. The final section presents the conclusion and future work.

2. RELATED WORK

Chaotic cryptography consists of two important paradigms: analog and digital chaotic systems [2]. The first paradigm is the analog chaotic communication systems. They are based on synchronization of two chaotic systems in a noisy environment [3]. Two chaotic systems can synchronize through coupling. One or more scalar signals are sent from one system to another or via a third external source. One of the advantages of the analog or free chaos is the ability to simultaneously encrypt and spread the signal. This minimizes devices and saves energy. Jamming attack is also difficult to apply to these systems [12]. However, analog chaotic systems face many deficiencies [12,13]:

- Two identical chaotic systems at least must be used to establish a communication. It is not evident to find two similar devices; synchronization therefore becomes more difficult.
- Tiny signals and transmission noise are difficult to differentiate. Synchronization noise must have lower intensity than the information signal.
- Synchronization time is hard to establish.

Moreover, the key space is reduced by the ineluctable errors of the components values and the signal redundancy. Many cryptanalyses are then possible [1,12]. For these different disadvantages, we are not going to consider analog chaotic systems in the rest of this paper. The second paradigm is the digital chaotic crypto-systems or ciphers, which are designed for digital computers. Chaotic maps are implemented in a finite precision [2]. We can classify digital chaotic crypto-systems into two main categories: chaotic stream ciphers and chaotic block ciphers.

2.1. Chaotic stream ciphers

Chaotic stream cipher encrypts each plaintext bit or byte one by one. It mixes the pseudorandom cipher bit stream, based on a chaotic map, with the plaintext bits using in general an Exclusive Logical OR. This type of chaotic cipher was introduced in1989 by Matthews when he presented a one-dimensional (1D) map with chaotic behavior [14]. Sequences of pseudorandom numbers are generated by this chaotic map and considered as a one-time pad [12,15]. We will not deal in detail with the different methods using stream chaotic ciphers because our study focuses on block ciphers.

2.2. Chaotic block ciphers

The block cipher encrypts blocks of bits with a length that differs according to the deployed algorithm. The security level of a chaotic block cipher is fixed by the properties and the implementation method of the chaotic function [13]. Several chaotic block ciphers proposed in the bibliography apply the Feistel structure [15–18]. Chaotic maps are employed to generate key streams or substitution table named S-Box. However, if we consider the methodology using the chaotic map, other classifications can be presented.

2.2.1. Chaotic block cipher for WSN

Recently, the interest of some researchers was focused on designing chaotic crypto-systems for WSNs. These chaotic crypto-systems challenge widely known algorithms such as RC5 [19], RC6 [20], and Advanced Encryption Standard (AES) [21]. Chen *et al*. proposed in [22] an algorithm based on the following 8-bit chaotic map:

$$y = (x \ll 2) - \left(x^2\right) \gg 6 - 1 \tag{1}$$

where x and y are 8-bit unsigned integers. The operators "$\ll$" and "$\gg$" are respectively right-bit shifting and left-bit shifting.

Chaotic properties of this map are not proved in [22]. The input right 4 bits are enlarged into 1 byte and XORed with the key k. The result is then iterated by the chaotic map quoted before. Then, an exclusive OR (XOR) operation is applied to the high 4 bits and low 4 bits. The Feistel structure is also used, but the number of rounds is too small, which is not consistent with the principle of Feistel structure. Some flaws of this algorithm allowed its cryptanalysis through differential attack [23].

Another chaotic security algorithm is presented in [24], and the authors proposed a new chaotic map called *N*-logistic–tent with an enlarged data range. The *N*-logistic–tent map is given by the following equations:

$$\begin{cases} x_{n+1} = \mu x_n (N - x_n/m)/N - y_n/2 \\ y_{n+1} = \beta (N - |N - y_n|) \end{cases} \tag{2}$$

where $x \in (0, m.N)$, $\mu \in (0,4)$, $y \in (0, 2*N)$, $\beta \in [1,2]$, $N = 2^K$, and $m = 2^k$ with integers K and k.

Dealing with integers simplifies the computation and minimizes the consumption. The authors suggest using 4 bytes to represent N, and choosing x_i and y_i between $(0, m*N)$ and $(0, 2*N)$ respectively. The seed key is the set (x_i, y_i, μ, β, m, N), and the data is XORed with the

chaotic keys. We used this map to generate chaotic S-Box as shown in Section 3.

2.2.2. Block cipher based on inverse chaotic systems

The proposed algorithms directly use the chaotic maps to encrypt messages. For example, Habutsu *et al*. [25] iterate the inverse of the skew tent map to encrypt information. Biham in [26] cryptanalyzed Habutsu algorithm using known-plaintext attack with 2^{38} complexity [15,18]. The skew tent map is piecewise linear, and in order to increase security, a more complex map with strong nonlinearities can be used. Masuda *et al*. [13] reused this map and expanded the range of x and α in order to use integers instead of real numbers. We are going to tackle this principle in a more detailed way, in the construction of S-Box.

2.2.3. Chaotic block cipher based on rounded chaos and S-Box

Another way to encrypt information using chaos is by means of chaotic S-Box. The nonlinear substitution table or S-Box represents the essential part of Feistel architecture. To construct an S-Box, designers rely not only on this categorization but also on exploiting chaotic pseudorandom generators and on inspiring from chaotic stream ciphers.

2.3. S-Box design: state of the art

Some of dynamic S-Box designs consist in iterating a chaotic map (one dimension or more) and discretizing output real values by different ways to create a matrix of integers between 0 and 255. They sometimes apply rows and columns rotation as well as permutations to enhance security of the S-Box. The second alternative relies on expanding data range of the chaotic map in use.

In this section, we will explain many designs of chaotic S-Box.

2.3.1. S-Box based on real output of chaotic map

We are going to present three principal ways to design S-Boxes using chaotic maps with real outputs.

The *maps used in this subsection have a data* output range [0,1] or $[-1, 1]$; so, the values are signed floats.

- Assign integers to subintervals of the phase space:

The first method was presented by Jakimoski in [15]. He proposed to discretize the Logistic map by dividing the phase space into $(n+1)$ equal intervals. After N iterations and if the output, which has a unique input, falls in the interval i, then assign to it the integer i.

Choose 256 inputs having a unique image and outputs, respectively, and then assign a new magnitude between 0 and 255 to these elements.

In [4], the authors inspire the same principle and reduce only the magnitude of n.

Using another chaotic map is also possible, and it was described in [5] where the authors iterate a piecewise linear chaotic map (PWLCM) to generate S-Box. The parameters' range of the PWLCM is continuous. All the values in this range guarantee the chaotic behavior, unlike the logistic map or the Henon attractor whose interval limits' definition is not simple because of the existence of periodic windows even where the chaos exists [2]. The PWLCM map has an exponentially decreasing autocorrelation and an invariant uniform density, which makes it suitable for chaotic crypto-systems.

This kind of map was employed by many others to create S-Box, and for instance, Wang *et al*. [6] conceived a four-step function $f(i,j,K)$ that generates a dynamic S-Box. The set (i,j,K) is considered respectively: number of iterations, subintervals number, and one of the random permutations of the sequence $\{0, 1, 2, \ldots, 2^n - 1\}$.

The steps of this function consist in dividing the phase space into j subintervals and iterating the chaotic map i times. The third step is to replace, in the sequence K, the nth integer k_n with the jth integer k_j if the chaotic map output is in the nth subinterval. The fourth step is to modify the control parameter b as stated by the following equation:

$$b = 0.9 + 0.1 * k_m/(2^n - 1) \tag{3}$$

To create i S-Box at the same time, the authors use i random permutation sequences: K_1, K_2, K_3, …, K_i, and repeat i times the function f.

- Binary sequence generation from real outputs:

The second method to construct chaotic S-Box suggested in [7] is to binarize the real outputs of the chaotic maps. To have integers between 0 and 255 from a real number, the idea is to obtain a binary sequence B_i^n and to convert each 8 bits into an integer. A float number x can be represented as [7]:

$$x = 0.b_1(x)b_2(x)b_3(x)\ldots b_i(x); x \in [0, 1] \tag{4}$$

the ith bit $b_i(x) \in \{0,1\}$ is defined as

$$b_i(x) = \sum_{r=1}^{2^i-1} (-1)^{r-1} \theta_{\left(r/2^i\right)}(x) \tag{5}$$

where $\theta_t(x)$ is the following threshold function:

$$\theta_t(x) = \begin{cases} 0, & \text{if } x < t \\ 1, & \text{if } x \geq t \end{cases} \tag{6}$$

The n-length binary sequence is

$$B_i^n = f(x) = \{b_i(\tau^n(x))\}_{n=0}^{\infty} \tag{7}$$

where τ is the nth iterations of the chaotic map. Each real output is binarized by applying the previous method, if the integer exists then the chaotic map is iterated once again. The S-Box is nonlinearly shuffled by applying a two-dimensional baker map. Chen *et al*. in [8] correct

some errors in Tang's paper [7] and enhance this method by increasing the dimension of the baker map (from two dimensional to three dimensional (3D)) and by using the Chebyshev map to generate the random binary sequence.

- Spatiotemporal chaos:

Yuan *et al*. in their paper [9] used spatiotemporal chaos to generate S-Box. The most used spatiotemporal chaotic system is the coupled map lattice:

$$x_{n+1}(i) = (1-\varepsilon)f(x_n(i)) + \frac{\varepsilon}{2}[f(x_n(i+1)) + f(x_n(i-1))] \tag{8}$$

where f is the logistic map used as a local map despite its drawbacks and n and i are respectively the time index and the lattice index. The initial lattice values are fixed, and the spatiotemporal system is iterated d times where $(d > 2N)$. Then, the lattice indexes are gathered in descending order from the maximum lattice to the minimum one. The maximum lattice value is chosen if two identical values occur. Finally, the S-Box $2^{n/2} \times 2^{n/2}$ is created.

2.3.2. S-Box based on integer outputs of chaotic map

Inspired by Masuda's method to discretize the tent map [13], Tang *et al*. proposed in [11] to adapt the algorithm in order to construct a dynamic S-Box. The skew tent map is a customized tent map obtained by choosing a critical point different from 0.5. It is defined by the following equation:

$$f_a(x) = \begin{cases} \dfrac{x}{a}; & \text{if } 0 < x \leq a \\ \dfrac{x-1}{a-1}; & \text{if } a < x \leq 1 \end{cases} \tag{9}$$

The discretized map is given by the next equation:

$$F_A = \begin{cases} \left\lceil \dfrac{M}{A} X \right\rceil; & \text{if } t < X \leq A \\ \left\lfloor \dfrac{M}{M-A}(M-X) + 1 \right\rfloor; & \text{if } A < X \leq M \end{cases} \tag{10}$$

The authors proposed to randomly choose the IC X_0 among the set $\{1,2,\ldots,256\}$ and then iterate the map more than k times. The iteration number must be equal to $2.39 \log_2 M + 15$ to satisfy chaotic properties.

Finally, we repeat the previous process until we have 256 elements to construct the S-Box. We suggest adding repetition test to be sure of not having duplicate values.

3. THE PROPOSED ALGORITHMS FOR S-BOX DESIGN

The principal concern of the presented S-Box design in the previous section was the security issue that was already proved by the authors. All the previous works are presented to enhance conventional ciphers. They do not take into consideration the complexity or the time spent to construct these S-boxes, and they are not designed for WSN security. Thus, we are motivated to develop robust S-Boxes with low power consumption, in order to be implemented on wireless sensor nodes.

To evaluate the energy consumption, by simulation and testbed platforms, for the S-Box design, we propose new methods based on the combined use of two chaotic maps with real and integer approaches.

3.1. A new S-Box based on real approach with two chaotic maps

To improve nonlinearity and immunity against differential attacks, cascading multiple chaotic structures seems to be interesting [1]. The cryptanalysis will be more complex as the output is established by many mixed chaotic orbits. Thus, we get the idea to combine two chaotic maps [27]–the first one is 1D [28], and the second one is 3D–in order to construct 16×16 S-Box. We proposed two methods [28]. In the first one, we iterate n times the 1D map after choosing randomly the ICs, and each three outputs is the input of the 3D map. Then, we iterate the 3D map N times and get the Nth iteration of (x_i,y_i,z_i) until having 256 different values. We sort all values, and each value is substituted by its rank [17]. We finally get a 16×16 S-Box. In the second method, we keep the same process but instead of substituting each value by its rank, we change the real values by their hexadecimal representation and then convert them into integers from 0 to 255. We repeat the procedure until we find 256 different random integers. Only one map can be used for the two ways to create dynamic S-Box. More details are given in the article [27]. Almost the same principle of sorting the chaotic outputs to get their rank is inspired in [10]. The chaotic map used is a continuous Lorenz map; then, solving this equation is required to obtain the orbits. The authors proposed to add rows and column shifting to the left and a column rotation so that the generated S-Box does not completely depend on the chaotic system.

3.2. A new S-Box based on integer approach with two chaotic maps

Chaotic algorithm speed as well as hardware and software feasibility are influenced by the use of floating point arithmetic. On the contrary, fixed-point arithmetic could improve the speed and guarantee the feasibility. Fixed-point arithmetic and simple chaotic systems are recommended to accelerate the encryption speed and ensure simple hardware and software implementation [29]. However, fixed-point arithmetic may lead to a leak of information. Short cycle length and convergence to a fixed point can probably occur.

The problem of short cycle length can inhibit chaotic cryptography transition from theory to practice. Tao *et al*.

in [30] suggested the injection of perturbation to reinforce chaotic cryptography. The modification will affect the orbit value or parameters or both at once.

The perturbation is periodic. It is triggered at $t = 0$. It can be performed using a linear feedback shift register (LFSR) or linear congruential generators.

To construct the S-Box, we used the discretized Lorenz map mentioned in [31] and given by the following equation:

$$\begin{cases} x_{k+1} = x_k + \sigma(y_k - x_k)\Delta t \\ y_{k+1} = y_k + [x_k(\rho - z_k) - y_k]\Delta t \\ z_{k+1} = z_k + [x_k y_k - \beta z_k]\Delta t \end{cases} \quad (11)$$

We choose the initial values x_0, y_0, and z_0, and we iterate the map n times where $n \geq 10$.

Applying the perturbation seems necessary like in [31] because of the apparition of redundancy after a number of iterations. So, the output of LFSR is XORed to y_{k+1} and z_{k+1}, whereas x_{k+1} is XORed to the integration step as expressed in the following equation [31]:

$$\begin{cases} y_{k+1} = y_{k+1} \oplus \text{LFSR} \\ z_{k+1} = z_{k+1} \oplus \text{LFSR} \\ \Delta t = \Delta t \oplus x_{k+1} \end{cases} \quad (12)$$

Contrary to the article [31], we limit the variables size and the integration step to 2 bytes (16 bits). The integration step is not static; it changes in each iteration. If the highest 4 bits of the LFSR are all zero, then the integration step is set to the highest LFSR byte, else it is set to the lowest byte. The highest and lowest bytes of the set $(x_{k+1}, y_{k+1}, z_{k+1})$ are stored in an S-table.

All previous steps are repeated until we have 256 different integers from 0 to 255.

We also used the map presented in [26] to generate a chaotic S-Box, but we choose to represent variables on 2 bytes only, where $m = 4$, $\mu = 4$, $N = 2^{14}$, and $\beta = 2$.

The double bytes representation has led to a convergence toward a fixed point. To remedy this problem, we add a perturbation via an LFSR.

In the next section, we examine the performances of the proposed S-Box to confirm their immunity especially against differential and linear cryptanalysis.

4. SECURITY ANALYSIS

The design of cryptographically good S-Boxes is based on essential criteria, which are as follows:

(1) Nonlinearity.
(2) Strict avalanche criterion (SAC).
(3) Equiprobable input/output XOR distribution.

To apply these criteria, the ICs for different S-Boxes are given in Table I.

Table I. Initial conditions of the proposed S-Boxes.

Proposed S-Box based on	Initial condition
Two chaotic maps (first method)	$p = 0.15$, $x_0 = 0.7$
Two chaotic maps (second method)	$p = 0.15$, $x_0 = 0.7$
N-logistic–tent map	$Y = 30000$, $x = 1200$
Discretized Lorenz map	$x = 0$, $y = 220$, $z = 6210$

4.1. Nonlinearity

Linear cryptanalysis consists in finding linear approximations of a cipher algorithm. The cryptanalysis tries to build linear equations connecting input plaintext with output encrypted text. For a given S-Box, all linear approximations are gathered in a linear approximation table. Each element of the table is based on the following expression:

$$L_p = \max_{a,b \neq 0} \left(\frac{\#\{x \in X, x \cdot a = f(x) \cdot b\} - 2^{n-1}}{2^{n-1}} \right)^2 \quad (13)$$

L_p is a linear approximation probability that measures the nonlinearity of a given function $f(x)$. # is the cardinality of the set x, $x \cdot a$ is the parity of the binary product of x and a, and $a, b \in \{1, 2, \ldots, 2^n - 1\}$.

In fact, each element of the table represents the number of equality between the linear equation of the input sum and the linear equation of the output sum. The result is then divided by the total number of possible inputs. The linear probability approximations of the S-Box are based on the two maps combination for the first and second methods (described in [27]), and the S-Box of the *N*-logistic–tent map are respectively 0.0625, 0.0705, and 0.0881. On the other hand, the Lorenz S-box L_p is 0.0976. These obtained values are close to the results in [7,8,15,17], which are respectively 0.0706, 0.0706, 0.0625 and 0.088135.

Hence, our proposed S-Boxes are secure against linear cryptanalysis because their linear probability approximations are among the lowest ones.

4.2. Strict avalanche criterion

Webster and Tavares in [32] introduced the SAC to test the S-Boxes. According to them, an f function satisfies this criterion if each 1-bit input changes half of each output bit change. Indeed, a dependence matrix might be calculated.

To calculate the dependence matrix, we produce first an m-bits cipher vector s from an n-random bits vector e. Two sets of n vectors $(e_1, e_1, \ldots, e_n)$ and $(s_1, s_2, \ldots, s_n)$ are then created, such that e_j and e differ only in 1 bit j, and $s_j = f(e_j)$.We calculate also a set of avalanche vectors $(v_1, v_2, \ldots, v_n)$ by applying an XOR between s and s_i. The next step is to add the ith bit of v_j to the element a_{ij} of the dependence matrix A. The previous step is repeated for k random vectors e, and each element of the matrix is divided by k. The resulting value a_{ij} varied from 0 to 1. A unit value means that the bit i should change each time the bit j is complemented. A half value means that the cryptographic function satisfies the SAC.

Table II. Dependence matrix of the S-Box based on two chaotic maps (first method).

0.4392	0.5490	0.4392	0.3765	0.4706	0.5020	0.4863	0.5020
0.4392	0.5176	0.5333	0.5647	0.5490	0.4549	0.4863	0.5804
0.5020	0.5333	0.5490	0.5490	0.4706	0.5176	0.5020	0.5333
0.4863	0.4549	0.4392	0.5804	0.5804	0.4235	0.4549	0.4863
0.4706	0.4863	0.5020	0.5804	0.4549	0.4706	0.5647	0.4235
0.5333	0.5490	0.4863	0.4863	0.5333	0.5647	0.5490	0.4549
0.4863	0.5804	0.4863	0.5176	0.5020	0.5176	0.4392	0.5333
0.5020	0.4392	0.4706	0.4863	0.5176	0.5176	0.4392	0.4549

Table III. Dependence matrix of the S-Box based on two chaotic maps (second method).

0.5804	0.4706	0.5490	0.5020	0.5333	0.5020	0.4549	0.5804
0.5020	0.4392	0.4863	0.5020	0.5020	0.4549	0.5176	0.4863
0.4706	0.5020	0.4863	0.5020	0.5020	0.5176	0.4863	0.4392
0.5176	0.6118	0.4863	0.5333	0.5020	0.5020	0.4392	0.4863
0.4078	0.4392	0.5333	0.5490	0.4706	0.5333	0.5647	0.5333
0.5490	0.5647	0.4706	0.5333	0.4706	0.5333	0.5490	0.4706
0.5333	0.4706	0.5176	0.4549	0.4863	0.5333	0.4549	0.4392
0.5647	0.5020	0.4706	0.4392	0.4549	0.5020	0.5176	0.4235

The corresponding dependence matrices of our proposed methods described earlier are given in Tables II–V. The mean value of each matrix is respectively 0.4993, 0.4998, 0.5105, and 0.5054, which are so close to the perfect value 0.5, so the S-Boxes satisfy the SAC. These values are equal or even better than the results in the following references [6,7,10,11,15] as well as the static S-Box. The mean values of their dependence matrices are respectively 0.5125, 0.4993, 0.5048, 0.4923, 0.4972, and 0.5069.This order is not the same as that in the paper.

4.3. Equiprobable input/output XOR distribution

Biham and Shamir introduced in [33] the differential cryptanalysis for an S-Box. It is based on the disproportion in the input/output XOR distribution table, so an S-Box resists the differential cryptanalysis attack if it has an equiprobable input/output distribution. The complexity of the differential attack is measured by the differential approximation probability of a map f (DP_f).

Table IV. Dependence matrix of the S-Box based on *N*-logistic–tent map.

0.5020	0.5333	0.5176	0.5020	0.5020	0.5647	0.4235	0.5176
0.5176	0.5333	0.4706	0.5020	0.5647	0.5020	0.4863	0.5020
0.4235	0.5020	0.6118	0.5490	0.4706	0.5490	0.5020	0.5490
0.4706	0.4706	0.5020	0.5804	0.5020	0.4863	0.4706	0.5333
0.5647	0.5490	0.5333	0.5020	0.5333	0.5020	0.5020	0.5490
0.4706	0.4863	0.4863	0.5020	0.4863	0.3922	0.4863	0.4706
0.4863	0.4863	0.5490	0.4706	0.4863	0.5804	0.4863	0.5647
0.5333	0.5490	0.5490	0.5647	0.4863	0.5176	0.5176	0.5176

Table V. Dependence matrix of the S-Box based on Lorenz map.

0.4549	0.5020	0.5176	0.5020	0.5647	0.4549	0.4706	0.5020
0.5490	0.4863	0.5333	0.5176	0.6118	0.5020	0.5333	0.4549
0.4863	0.5176	0.5176	0.5020	0.4392	0.5804	0.4549	0.5647
0.5176	0.5647	0.5490	0.5333	0.4549	0.5020	0.5176	0.5333
0.5020	0.5020	0.5490	0.5020	0.5020	0.5020	0.5490	0.4863
0.5490	0.4235	0.5333	0.4863	0.4706	0.4706	0.4863	0.5176
0.5176	0.4392	0.4392	0.5333	0.5333	0.4863	0.5020	0.4549
0.5333	0.5333	0.4235	0.5020	0.5490	0.5020	0.4706	0.5020

Table VI. Differential approach table of the S-Box based on two chaotic maps (first method).

–	8	6	6	10	6	6	6	6	6	6	8	6	6	8	6
6	8	8	6	6	8	6	6	6	6	6	4	8	6	6	8
6	6	8	6	8	8	8	6	6	8	6	6	10	6	8	6
8	8	4	6	8	8	8	6	6	6	6	6	6	6	8	8
6	8	6	8	6	6	6	6	6	6	6	6	8	8	6	6
8	6	6	8	8	8	6	8	6	10	6	6	8	6	8	8
6	8	6	4	6	6	8	6	6	6	6	6	6	8	6	6
6	6	6	6	8	8	8	8	6	6	6	6	8	6	8	6
6	8	6	10	8	6	8	6	10	6	8	6	6	6	8	8
4	8	6	6	6	6	6	8	6	6	6	6	6	8	6	6
6	6	6	6	10	6	8	6	8	6	8	6	6	6	8	8
6	8	8	6	8	8	6	6	8	8	6	6	8	8	6	8
4	6	6	6	6	6	8	10	6	8	6	6	8	6	6	6
6	8	8	8	10	6	6	6	8	8	6	8	6	8	8	6
6	10	6	8	6	8	6	6	6	8	6	10	6	8	8	8
8	8	6	6	6	6	6	6	6	6	6	6	6	8	8	6

Table VIII. Differential approach table of the S-Box based on *N*-logistic–tent map.

–	8	8	6	8	6	6	6	6	8	6	10	8	6	6	6
8	6	4	6	8	6	8	6	6	6	6	6	8	6	6	6
6	6	6	8	6	6	6	8	8	8	6	6	6	6	6	6
8	6	6	6	8	8	6	8	6	6	6	8	8	6	8	6
8	8	6	10	8	8	6	6	8	6	6	8	8	8	6	8
10	6	6	6	6	8	6	8	8	8	8	8	8	8	6	8
8	8	6	8	6	6	6	6	6	6	8	6	10	6	6	8
8	6	6	8	8	6	6	6	6	6	6	6	6	6	8	8
6	6	8	6	6	6	8	8	6	8	6	6	6	8	6	6
6	6	6	8	6	6	6	4	6	8	6	6	6	8	6	8
6	6	8	6	8	6	8	6	6	6	10	6	6	6	6	8
6	8	8	8	6	8	6	6	6	6	6	6	6	6	8	8
8	6	8	6	6	6	6	6	8	6	8	6	6	6	6	6
8	10	8	10	6	6	6	6	6	6	6	8	6	6	8	6
6	6	8	8	8	8	6	10	6	8	8	4	6	6	6	6
6	6	6	6	8	6	8	6	6	6	6	6	6	6	6	8

$$D_p = \max_{\Delta x \neq 0, \Delta y} \left(\frac{\#\{x \in X, f(x) \oplus f(x \oplus \Delta x) = \Delta y\}}{2^n} \right) \quad (14)$$

where 2^n is the cardinality of all the possible input values (x), Δx is the input differential, and Δy is the output differential.

The occurrence frequency of the most probable output XOR is represented by the differential approach table.

Tables VI–IX represent differential approach tables of the proposed S-Boxes. We notice that the DP_f of each table is 10/256, which is equal to the results in the following articles [5–7,10,11]. The DP_f in Jakimoski's [15] and Chen's [8] papers is 12/256.

Thus, the proposed S-Boxes have equiprobable input/output XOR distribution and could resist differential attack.

Security analysis of our S-Boxes shows how good they are. In fact, our approaches almost satisfy the criteria for a cryptographically secure S-Box, so they can resist linear and differential attacks with similar or even better performances than the existing propositions.

The current results encourage us to propose an implementation of these dynamic chaotic S-Boxes for secure WSN.

5. IMPLEMENTATION ON WIRELESS SENSOR NETWORK

We implement our chaotic dynamic on WSN to secure information. In our case, dealt with in this section, we implement the AES algorithm and a modified AES algorithm where we use a dynamic S-Box instead of the static one. The challenge is to design an approach that fulfills the

Table VII. Differential approach table of the S-Box based on two chaotic maps (second method).

–	6	6	8	8	8	8	6	6	8	8	6	8	6	8	6
6	8	6	6	6	6	8	6	6	6	6	6	6	10	10	6
6	6	6	6	6	6	6	8	8	8	6	8	8	6	6	8
4	6	6	6	8	6	6	6	6	8	6	8	6	6	6	6
6	6	6	8	8	6	6	6	8	8	8	8	6	8	8	6
6	6	6	8	6	6	6	6	6	6	6	10	6	8	6	6
8	6	6	6	8	6	8	6	8	6	6	6	8	6	6	6
6	8	8	6	6	8	8	6	8	8	8	6	8	10	8	6
6	6	6	8	8	6	6	4	6	8	6	6	8	6	6	6
8	8	6	6	6	6	8	6	6	8	6	4	6	8	6	6
6	6	8	6	6	6	6	8	6	6	6	8	8	8	6	6
6	6	8	4	6	6	6	6	6	10	6	6	6	8	6	8
6	6	8	6	6	6	8	8	6	6	8	6	8	6	6	6
10	6	6	6	8	6	8	6	8	6	8	8	6	10	10	6
8	6	6	6	8	10	8	6	6	6	6	6	8	6	6	6
10	6	6	6	8	8	6	6	6	8	8	6	8	6	8	6

Table IX. Differential approach table of the S-Box based on Lorenz map.

–	6	8	6	6	6	8	8	6	6	6	8	6	8	8	6
6	8	8	8	6	6	8	8	8	6	6	6	4	8	8	6
8	8	8	6	6	10	8	8	6	6	6	10	6	8	6	8
6	8	6	6	8	6	6	8	6	6	8	6	6	6	8	6
6	6	10	6	8	6	6	6	6	6	6	6	6	6	6	8
8	8	6	6	6	6	6	8	6	8	6	8	6	8	6	8
6	6	6	6	6	6	6	6	10	6	6	6	6	8	8	8
6	6	10	10	10	8	8	6	6	8	6	6	8	6	6	6
8	8	6	6	6	6	6	8	6	8	8	10	8	6	6	6
6	6	6	6	6	6	6	6	8	8	6	6	6	10	6	6
6	4	6	6	6	6	8	6	6	6	8	6	6	6	6	6
6	8	6	6	6	6	6	8	6	6	6	6	8	6	8	8
6	6	6	6	8	6	8	6	6	6	6	6	10	6	6	8
6	8	8	6	6	6	8	8	4	6	6	8	6	6	8	4
6	4	6	8	6	8	6	6	6	6	8	6	6	4	8	8
8	6	8	8	8	8	8	6	6	6	6	6	6	6	8	6

requirements of such network and to propose a communication protocol in order to guarantee the synchronization of the dynamic S-Box generation.

Wireless sensor network is based on small sensor nodes with limited energy, memory, and computational resources [34–36]. Energy consumption is very critical because the nodes are equipped with limited lifetime battery and replacing them is not always an option. These limitations make security services, especially computationally complex cryptographic algorithms, more difficult to implement in WSNs.

For this purpose, we used WSim [37] and WSNet [38] simulators to test the proposed S-Boxes and compare their performance especially concerning the energy consumption.

We carry out our codes on WSim MSP430 platform simulator, which is equipped with an MSP430f1611 micro-controller and a TI CC1100 radio chipset. A 16-byte plaintext is encrypted and then transmitted to other nodes with a token ring protocol.

As seen in Section 2.3, designing chaotic S-Box is arranged in two categories. The first one deals with real outputs of chaotic maps, and the second one is related to integer chaotic map outputs. As we said before, there is no implementation of dynamic chaotic S-Box in WSN, so we implement only two principal proposed designs that we found in the literature to compare their performance with our proposed methods. For the first category, we chose to implement the assignment of integers to the phase space subintervals by dividing intervals. It seems simpler and adequate to WSNs. Even though there are many proposed methods in the literature, they all have the same principle. For this reason, we implement a unified model given in the following pseudo-code:

Algorithm 1 S-Box construction

1: $x_1 \leftarrow$ Initial Condition

2: **while** $j<Max_{iteration}$ **do**

3: $x_2 \leftarrow$ PWLCM (x_1)

4: $i \leftarrow$ DICHOTOMY (x_2)

5: $SBox[j] \leftarrow i - 1$

6: **end while**

For simplicity and consumption reasons, we chose to restrict the dimensions of the used maps to a 1D PWLCM map.

To simplify integer values assignment of each real chaotic map output, we choose the assignment by dichotomy or binary search (Method 2). The binary search function halves the search space, refines the search by dividing intervals by multiples of two, and tests the existence of the real value (output) in the interval in question. If it is true, the function returns the interval position. The maximum number of tests is the first exponent of 2 greater than or equal to the possible intervals number N. It can be defined by the following equation:

$$M = (\log_2(N)) \tag{15}$$

If we do not have repetitions, the maximal number of intervals is 256. We will apply eight tests to each chaotic output value. Given that we have 256 values, the total number of tests is 2048.

Figure 1 shows the iteration number that we need to construct the S-Box for a specific IC. The mean value is 2376, which indicates that many states fall in the same intervals and generate repetitions.

We also calculate the occurrences of each iteration number that we illustrate by Figure 2.

In this figure, the iteration numbers are gathered by intervals of 100. The most probable iteration numbers are between 1400 and 1500. This value represents only 28% of the total.

The binary sequence generation from real outputs proposed in [7], when implemented, appears very computational in resources; so, we only test the binarization function, and every real output of a PWLCM map is transformed via Equation (5) to 8 bits.

We also implement our approach based on sorting the array and getting the rank of each output (Method 1). The control parameters and the ICs form a part of the modified system key. The sort function is the most complex one in our algorithm. We choose a fast and non-recursive sorting algorithm, which is the well-known Combsort algorithm. For 10^6 tested ICs, there are no repetitions of the outputs. Therefore, for any ICs, we get the S-Box after 256 iterations.

On the other hand, in the second category (integer approach), we implement three methods: the S-Box proposed by Tang in [11] based on the discretized skew Tent map ($Method_3$) and our contributions constructed respectively with the discretized Lorenz map (Method 4) and the N-logistic–tent map (Method 5). The pseudo-code of the S-Box based on the discretized Lorenz map is given in Algorithm2. For both approaches, we used a 16-bit Fibonacci LFSR.

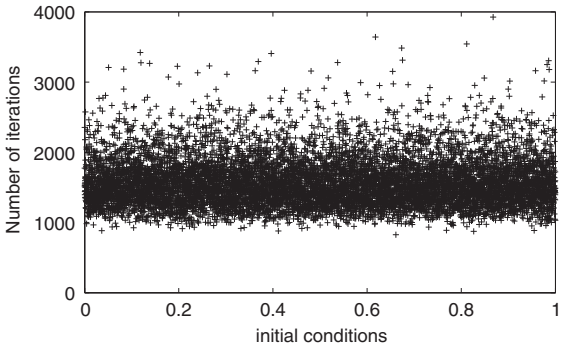

Figure 1. Iteration number depending on initial conditions with 0.0001 step.

Algorithm 2 S-Box generation with Lorenz method

$x_0, y_0, z_0 \leftarrow$ Initial condition
$j \leftarrow 0$
$\text{LFSR}_{seed} \leftarrow$ 0xACE1u
while $j<256$ **do**
  $x_{j+1}, y_{j+1}, z_{j+1} \leftarrow \text{LORENZ}\,(x_j, y_j, z_j)$ mod 0xFFFF
  if (j mod $1 == 0$) **then**
    $y_{j+1} \leftarrow y_{j+1} \oplus \text{LFSR}_j$
    $z_{j+1} \leftarrow z_{j+1} \oplus \text{LFSR}_j$
  end if
  $\Delta_t \leftarrow \Delta_t \oplus x_{j+1}$
  if ($4*$highest bits of $\text{LFSR}_j == 0$) **then**
    $\Delta_t \leftarrow 1*$highest byte of LFSR_j
  else
    $\Delta_t \leftarrow 1*$lowest byte of LFSR_j
  end if
  $S_{xh} \leftarrow 1*$highest byte of x_{j+1}
  $S_{xl} \leftarrow 1*$lowest byte of x_{j+1}
  $S_{yh} \leftarrow 1*$highest byte of y_{j+1}
  $S_{yl} \leftarrow 1*$lowest byte of y_{j+1}
  $S_{zh} \leftarrow 1*$highest byte of z_{j+1}
  $S_{zl} \leftarrow 1*$lowest byte of z_{j+1}
  $SBox[j] \leftarrow S_{xh} \parallel S_{xl} \parallel S_{yh} \parallel S_{yl} \parallel S_{zh} \parallel S_{zl}$
end while

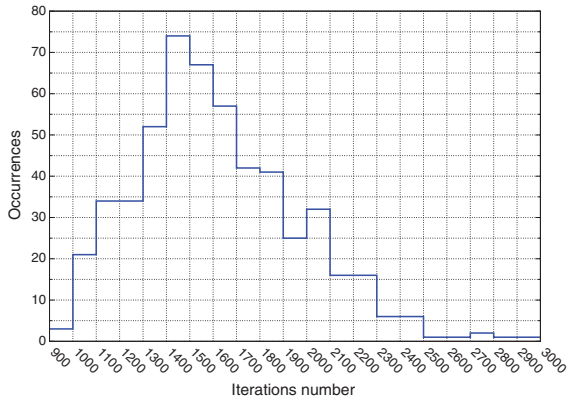

Figure 2. Occurrences of iteration number interval.

In order to find the number of iterations needed to construct the S-Boxes for different ICs, we try to do an exhaustive search, which appears impossible to us because of the huge number of possibilities of the ICs (about 2^{48} combinations for the S-Box based on discretized Lorenz map and 2^{32} for the S-Box based on *N*-logistic–tent map). We divide the intervals and sweep the values with a variable step, and then, we find the minimal iteration number and their occurrences.

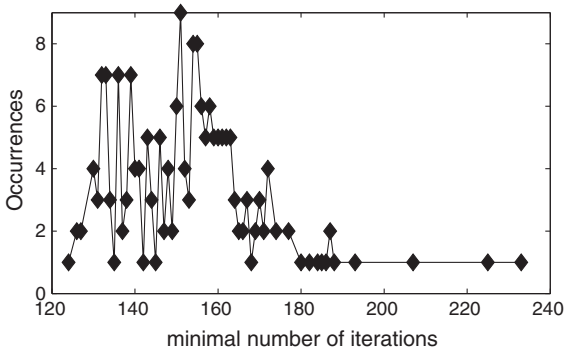

Figure 3. Occurrences of minimal iteration numbers with different initial conditions for Lorenz map.

Figure 3 illustrates the occurrences of minimal iteration numbers for different ICs of the discretized Lorenz map S-Box.

The most probable minimal iteration number is 151. The lowest and the highest numbers of iterations are respectively 124 and 233, which occur only once.

Figure 4 shows the occurrences of the minimal iteration number of the *N*-logistic–tent map S-Box. For different ICs, 320 is the most probable minimal number of iterations. After defining the implemented codes and counting the iteration numbers of each one, we notice that some of our codes lead to obtain 16×16 S-Boxes after more than 2^8 iterations. The complexity of the algorithms and the iteration numbers strongly affect the energy consumption evaluated in the next section.

6. ENERGY CONSUMPTION

In this section, we present, in details, the simulation and the experimental results about energy consumption of our different proposed S-Boxes with the IC presented in Table I. For the S-Boxes based respectively on discretized

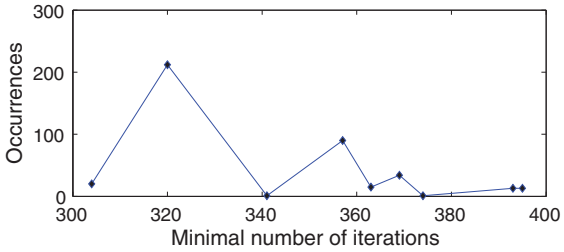

Figure 4. Occurrences of minimal iteration numbers with different initial conditions for *N*-logistic–tent map.

Lorenz map and *N*-logistic–tent map, these ICs match with the highest number of iterations.

6.1. Simulation results of energy consumption

The energy profile is estimated via eSimu. eSimu is an energy consumption simulator for WSN nodes. It estimates energy profile via WSim traces [39,40].

Figure 5 illustrates the energy consumption (10^{-4} J) of five approaches. The first and the second one (Method 1 and Method 2, respectively) are based on real output chaotic maps. Dividing the phase space with dichotomy (Method 2) is not suitable to assign integer values to the chaotic state. Sorting the outputs (Method 1) appears less energy consuming with 63.726×10^{-4} J, which is a static value for all the ICs.

The three last approaches (Method 3, Method 4, and Method 5) are based on chaotic maps with integer outputs. The S-Box based on Lorenz map (Method 4) has the lowest consumption rate with only 1.92234×10^{-4} J. However, Method 3 proposed in [11] consumes 779.338×10^{-4} J for $K_{\min} = 35$.

The consumption of the fifth approach (Method 5) based on *N*-logistic–tent map is close to the first one with 59.397×10^{-4} J.

Intuitively choosing a discretized chaotic map is less computational and more suitable for WSN implementation because of the fixed-point representation simplicity and speed.

However, Method 3 consumes more than the real chaotic map approaches. The discretized skew tent map relies on relatively complex operations such as division, ceil, and floor operations as well as the threshold function, which is also the binary search operation base function.

The choice of the chaotic map, the simplicity of the algorithm, and the adequate arithmetic representation are determinant to implement a low-cost chaotic S-Box.

The second part of this section is to compare the energy consumption that adds the generation of dynamic chaotic S-Box when used in the AES algorithm instead of the static one.

Figure 6 shows the energy consumption of each AES function needed to encrypt 16 bytes with 128-bit AES. We notice that the "mixsubcolumn" and "inv-mixsubcolumn" are the most consumer AES functions. The "mixsubcolumn" gathers the "subbyte" and "mixcolumn" functions. It uses the S-Box to substitute each byte of the state matrix and then multiplies the resulted columns with a fixed polynomial. On the other hand, the "inv-mixsubcolumn" regroups the "inv-subbyte" and the "inv-mixcolumn". The following pseudo-codes represent a portion of the "mixsubcolumn" and the "inv-mixsubcolumn" lines of code.

Algorithm 3 Mixsubcolumn portion of code

1: $S'[0] \leftarrow Sboxtimes2[state[0]] \oplus Sboxtimes3[state[5]] \oplus Sbox[state[10]] \oplus Sbox[state[15]]$

/* binary polynomial modulo multiplication of polynomial matrix and the state shifted and subtituted matrix */

/* Sboxtimes2 and Sboxtimes3 are precalculated matrices which constitute the modular multiplication of each S-Box column value with 2 and 3 respectively. */ /* Each column of the state matrix is substituted and mixed */

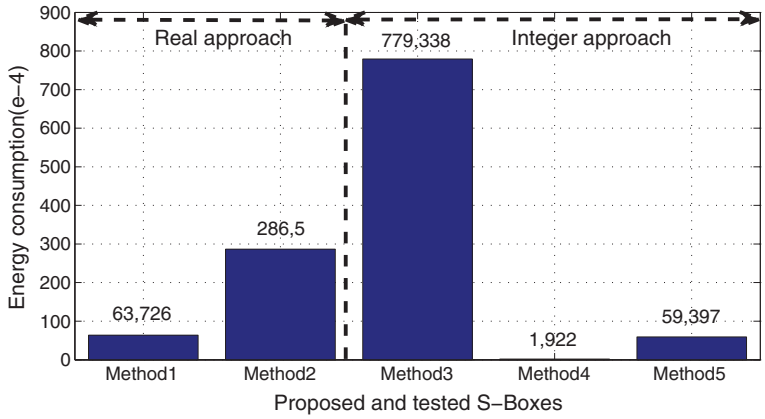

Figure 5. Energy consumption of proposed and tested S-Boxes.

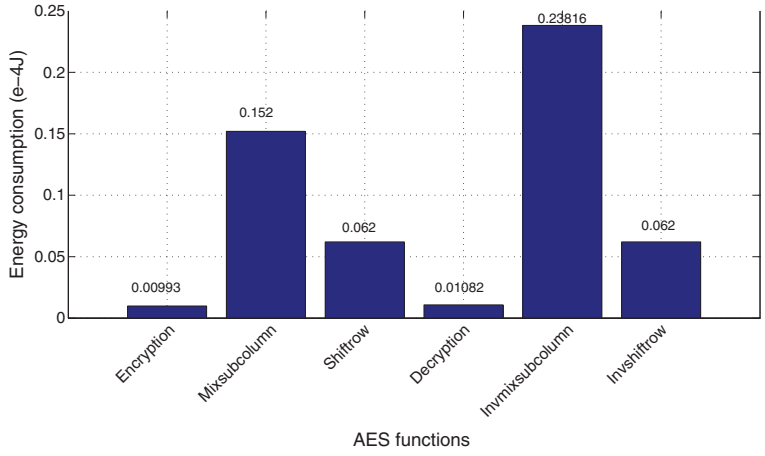

Figure 6. Energy consumption of AES functions.

Algorithm 4 Inv-Mixsubcolumn portion of code

1: **for** $i = 0$ **to** $4 * Nb$ **do**

2: $state[i] \leftarrow InvSbox[S'[i]]$

3: **end for**

The first line given in the "Mixsubcolumn" consumes 0.00925×10^{-4} J, and it is called 16 times to construct the new state matrix. The total consumption is then 0.148×10^{-4} J. The "inv-mixsubcolumn" adds a loop function (shown in Algorithm 4), which consumes 0.0876×10^{-4} J. It explains the energy difference between the two AES functions. On the other side, Figure 7 presents the surplus of energy consumption of the modified AES when compared with the static one. Despite the low energy cost of certain dynamic S-Box design methods, the energy surplus (4.31×10^{-4} J for the best one) is still a drawback that the dynamic aspect adds to the static AES algorithm. We can compensate this disadvantage by reducing the size of the S-Boxes and increasing the frequency of their generation.

In order to validate the simulations results, we manage to do some experiments with "real-world" conditions.

6.2. Experimental measurements of energy consumption

The experiments were realized on SensLAB platform, which is a very large-scale open WSN testbed [41]. It contains 1024 fixed and mobile sensor nodes deployed on four different sites with a distributed topology. SensLAB allows interacting with sensors such as real-time energy consumption measurement, radio activity monitoring, and data retrieval in real time. Each sensor has a TI MSP430f1611 micro-controller with a 16-bit processor running at 8 MHz. The radio interface is TI CC1101 (open MAC protocol) or CC2420 (Zigbee/IEEE 802.15.4) with a frequency of 868 MHz and 2.4 GHz, respectively. We flash our firmwares associated with the configuration parameter

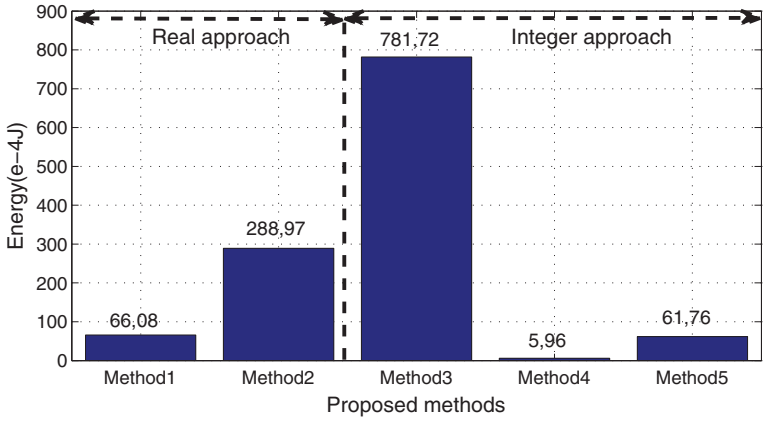

Figure 7. Surplus of energy consumption of the modified AES.

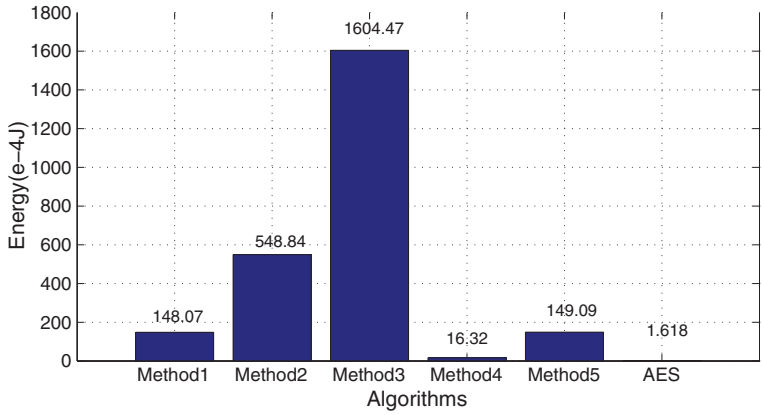

Figure 8. Energy consumption of proposed algorithms on SensLAB platform.

sets (called profile) on SensLAB nodes. The profile includes a configuration of node monitoring measurement and a power supply mode. Figure 8 represents the energy consumption measurements of each code simulated in the previous subsection. The experimental results show that Method 4 and Method 3 are respectively the lowest and highest chaotic algorithms, with energy consumption of 16.32×10^{-4} J and 1604.47×10^{-4} J, respectively. These measurements match the eSimu results in terms of algorithms consumption order but seem to be more consuming compared with the simulations. These differences can be explained by the simulator energy models approximations that can affect especially fast algorithms.

7. CONCLUSION

In this paper, we give a state of the art of the chaotic crypto-systems, with special focus on the dynamic chaotic S-Box design. The S-Box is considered as the core of many well-known algorithms such as Data Encryption Standard and AES. Many studies on algebraic attacks are carried out in order to write these S-Box with few and simple algebraic relations. The chaos theory provides a way to strengthen encryption algorithms or to design a new one totally based on chaos. We propose some chaotic S-Box design that we classify into two categories: S-Box based on real output of chaotic map and S-Box based on integer outputs of chaotic maps. We demonstrate that the proposed designs satisfy good S-Box criteria and can resist popular cryptanalysis: differential and linear attacks.

The final part of this paper describes the implementation of the dynamic S-Box, which replaces the static AES S-Box on WSNs. Some implementation requirements are proposed to optimize intuitively the time and the energy cost that we studied with eSimu.

Each proposed algorithm is implemented on WSim MSP430 platform simulator, which is equipped with MSP430f1611 micro-controller models with only 16-bit operations.

For WSN, using floating point seems to be inadequate. Hence, using discretized maps and limiting the outputs to only 16 bits (2 bytes) are deemed necessary. Unfortunately, they affect the chaotic properties of the discretized Lorenz and *N*-logistic–tent maps in that they generate short cycle lengths and repetitions. To remedy these problems, we inject a perturbation via a linear feedback register.

We found that although modular or free chaos adds a dynamic aspect and robustness, it increases AES encryption/decryption functions' energy consumption. Simulations showed that the energy surplus of the best S-Box design method is 4.31×10^{-4} J. The idea is to reduce the size of the S-Box and increase the frequency of its generation in order to reduce energy consumption.

An implementation on real WSN platform SensLAB as well as experimental energy measurements are carried out. The experimental results are approximately twice as much as the simulations, which means that WSim models are more optimistic and do not fit fast algorithms. Future work will focus on a self-organized protocol based on dynamic S-Boxes and a fully chaotic crypto-system for WSNs.

REFERENCES

1. Silva RM, Crespo RG, Nunes M. Loba128, a Lorenz based PRNG for wireless sensor networks. *International Journal of Communication Networks and Distributed Systems* 2009; **3**(4):301–318.
2. Alvarez G, Li S. Some basic cryptographic requirements for chaos-based cryptosystems. *International Journal of Bifurcation and Chaos* 2006; **16**: 2129–2151.
3. Pecorra L, Carrol T. Synchronization in chaotic systems. *Physical Review Letters* 1990; **64**:821–824.
4. Ou CM. Design of block ciphers by simple chaotic functions. *IEEE Computational Intelligence Magazine* 2008; **3**(2):54–59.

5. Asim M, Joeti V. Efficient and simple method for designing chaotic S-Boxes. *ETRI Journal* 2008; **30**(1):170–172.
6. Wang Y, Wong KW, Liao X, Xiang T. A block cipher with dynamic S-boxes based on tent map. *Communications in Nonlinear Science and Numerical Simulation* 2009; **14**(7):3089–3099.
7. Tang G, Liao X, Chen Y. A novel method for designing S-boxes based on chaotic maps. *Chaos, Solitons and Fractals* 2005; **23**:413–419.
8. Chen G, Chen Y, Liao X. An extended method for obtaining S-boxes based on three-dimensional chaotic baker maps. *Chaos, Solitons and Fractals* 2007; **31**:571–579.
9. Yuan H, Luo L, Wang Y. An S-box construction algorithm based on spatiotemporal chaos. *International Conference on Communications and Mobile Computing*, 2010.
10. Ozkaynak F, Ozer AB. A method for designing strong S-Boxes based on chaotic Lorenz system. *Elsevier Physics Letters A* 2010; **374**:3733–3738.
11. Tang G, Liao X. A method for designing dynamical S-boxes based on discretized chaotic map. *Chaos, Solitons and Fractals* 2005; **23**:1901–1909.
12. Alvarez G, Montoya F, Romera M, Pastor G. Chaotic cryptosystems. *Proceedings IEEE 33rd Annual 1999 International Carnahan Conference on Security Technology*, 1999; 332–338.
13. Masuda N, Aihara K. Cryptosystems with discretized chaotic maps. *IEEE Transactions on Circuits and Systems* 2002; **49**(1):28–40.
14. Matthews R. On the derivation of a chaotic encryption algorithm. *Cryptologia* 1989; **13**(1):29–42.
15. Jakimoski G, Kocarev L. Chaos and cryptography: block encryption ciphers based on chaotic maps. *IEEE Transactions on Circuits and Systems* 2001; **48**(2): 163–169.
16. Chen S, Zhong X. Confidential communication through chaos encryption in wireless sensor network. *Journal of China University of Mining & Technology* 2007; **17**(2):258–261.
17. Benjeddou A, Taha A, Fournier-Prunaret D, Bouallegue R. A new cryptographic hash function based on chaotic S-Box. In *CSNDSP*. IEEE Communication Systems Networks and Digital Signal Processing: Graz, Austria, 23-25 July, 2008.
18. Xiang T. A novel symmetrical cryptosystem based on discretized two-dimensional chaotic map. *Physics Letters A* 2007; **364**:252–258.
19. Rivest RL. The RC5 encryption algorithm. *Technical Report*, MIT Laboratory for Computer Science, 1995.
20. Rivest RL, Robshaw MJB, Sidney R, Yin YL. The RC6 block cipher. *Technical Report*, MIT Laboratory for Computer Science, 20 August 1998.
21. (FIPS) FIPSP. Advanced Encryption Standard (AES). *Technical Report*, PUB. 197, Nov 2001.
22. Chen S, Zhong X, Wu Z. Chaos block cipher for wireless sensor network. *Science in China Series F: Information Sciences* 2008; **51**(8):1055–1063.
23. Yang J, Xiao D, Xiang T. Cryptanalysis of a chaos block cipher for wireless sensor network. *Communications in Nonlinear Science and Numerical Simulation* 2011; **16**(2):844–850.
24. Fang Q, Liu Y, Zhao X. A chaos-based secure cluster protocol for wireless sensor networks. *Kybernetika* 2008; **44**(4):522–533.
25. Habutsu T, Nishio Y, Sasase I, Mori S. A secret key cryptosystem by iterating a chaotic map. *Proc. Advances in Cryptology EUROCRYPT'91*, Berlin, Germany, 1991; 127–140.
26. Biham E. Cryptanalysis of the chaotic-map cryptosystem. *Proc. Advances in Cryptology EUROCRYPT'91*, Berlin, Germany, 1991; 532–534.
27. Zaibi G, Peyrard F, Kachouri A, Fournier-Prunaret D, Samet M. A new design of dynamic S-Box based on two chaotic maps. *ACS/IEEE International Conference on Computer Systems and Applications*, Hammamet, Tunisia, 2010.
28. Zaibi G, Peyrard F, Kachouri A, Fournier-Prunaret D. On dynamic chaotic S-Box. *GIIS'09 Proceedings of the Second International Conference on Global Information Infrastructure Symposium*, 2009.
29. Bose R, Pathak S. A novel compression and encryption scheme using variable model arithmetic coding and coupled chaotic system. *Transactions on Circuits and Systems* 2006; **53**(4):848–857.
30. Tao S, Ruili W, Yixun Y. Perturbance-based algorithm to expand cycle length of chaotic key stream. *Electronics Letters* 1998; **34**(9):873–874.
31. Silva RM, Crespo RG, Nunes M. Enhanced chaotic stream cipher for WSNs. *IEEE International Conference on Availability, Reliability and Security*, 2010.
32. Webster A, Tavares SE. On the design of S-Boxes. *Proceeding Lecture notes in Computer Sciences; 218 on Advances in Cryptology CRYPTO'85*, 1989.
33. Biham E, Shamir A. Differential cryptanalysis of DES like cryptosystems. *CRYPTO'90 & Journal of Cryptology* 1991; **4**(1):3–72.
34. Anastasi G, Conti M, Francesco MD, Passarella A. Energy conservation in wireless sensor networks: a survey. *Ad Hoc Networks* 2009; **7**:537–568.
35. Li G, Ling H, Znati T, Wu W. A robust on-demand path-key establishment framework via random key predistribution for wireless sensor networks. *Hindawi Publishing Corporation EURASIP Journal on Wireless Communications and Networking* 2006; **2006**(2):1–10.

36. Kaps JP, Sunar B. Energy comparison of AES and SHA-1 for ubiquitous computing. *Lecture Notes in Computer Science* 2006; **4097**:372–381.
37. WSim: a software-driven simulator for full platform estimations and debug. URL http://wsim.gforge.inria.fr
38. WSNet simulator. URL http://wsnet.gforge.inria.fr/
39. Fournel N, Fraboulet A, Feautrier P. eSimu: a fast and accurate energy consumption simulator for real embedded system. *WOWMOM'07*, 2007; 1–6.
40. eSimu simulator. URL http://esimu.gforge.inria.fr
41. SensLAB Website. URL http://www.senslab.info